УДК 004.031.42

ОПЫТ РАЗРАБОТКИ WEB-ПРОЕКТА НА ЯЗЫКЕ PHP И СУБД MYSQL

Щербаков А.С., Теплинский С.В.
Донецкий национальный технический университет

Web-разработки сегодня применяются практически во всех сферах деятельности, при этом создание и размещение в глобальной сети web-сайта доступно всем желающим. Создание простых (статичных) сайтов не вызывает проблем в обеспечении их безопасности и защиты от взлома. Однако большие сложности возникают при разработке интерактивных систем, которые принимают от пользователя и обрабатывают различные данные.

Наибольшей популярностью среди web-разработчиков пользуется свободно распространяемый язык программирования PHP благодаря его большой гибкости и функциональности. По тем же причинам из многих СУБД предпочтение отдают MySQL.

Из типичных ошибок разработчиков web-сайтов можно выделить следующие:

- Синтаксические и логические ошибки в программном коде.
- Ошибки и уязвимости, позволяющие несанкционированно манипулировать контентом сайта (доступность SQL-Injections, возможность вставки скриптов или некорректных стилей на страницы форумов и гостевых книг, XSS – cross site scripting).
- Некорректное отображение и функционирование сайта в различных браузерах, в том числе браузерах, которые не поддерживают отображение графики, выполнение скриптов, или в которых эти функции отключены.

Поставлена задача разработать сайт, избегая типичных ошибок и сделать его максимально неуязвимым к хакерским атакам. Чтобы избежать возникновения таких ошибок во время разработки следует:

- Многократно тестировать программный код по мере его написания
- Считать, что запросы к серверным скриптам формируются не только имеющимися страницам и формами на них, но и потенциальным взломщиком. В случае реализации проверки вводимых пользователем данных на клиентской стороне, необходимо проверять их и на сервере – обойти клиентскую проверку очень легко.
- Выбирать оформление и разметку сайта такими, чтобы в браузерах с минимальными возможностями сайт отображался корректно и с минимальными потерями функциональности. При использовании скриптов, выполняемых на стороне клиента, предусмотреть бесскриптовую версию сайта.

В качестве примера в данном докладе приведен разработанный автором интерактивный сайт, на момент написания доклада доступный по адресу http://sp06n.ds8.ru. Сайт разрабатывался ориентированным на минимизацию уязвимостей.

Распространенной уязвимостью является уязвимость на SQL-инъекции. Такой метод взлома реализуется вставкой SQL-запросов в строку URL браузера вместо

определенного параметра, если серверная программа (скрипт) не проверяет корректность принятых данных, то этот запрос может быть выполнен в СУБД.[2] Например, проверку логина и пароля пользователя можно выполнить следующим запросом: *select * from users where login='$login' and pasw='$password'*. Если переменная $login не проверяется на некорректные символы, то пользователь может указать в качестве логина строку вида *Admin' --* В результате в СУБД будет передан запрос *select * from users where login=' Admin' --' and pasw='qwerty'*. Символы -- в MySQL являются комментарием, поэтому строка после них будет проигнорирована и пользователь выполнит вход с правами администратора. Чтобы этого избежать, необходимо проверять все входные данные, проверки лучше реализовать по типам. Если все запросы к СУБД строить в виде *select <поля> from <таблица> where field='value'* , то достаточно: в строковых параметрах удалять или заменять на эквивалент символ одинарной кавычки и символ обратного слеша (можно заменить на \\), из числовых параметров удалять все символы, не являющиеся цифрами, для паролей использовать хеш (это позволит избежать ограничений на использование кавычки и сделает практически невозможной утечку паролей в случае взлома или нечестности модераторов). В разработанных скриптах для проверки данных реализованы такие функции:

```
function prepare  login($data) // подготовка строки
        {return str  replace('\'','&rsquo;',str  replace('\\','\\\\',$data));}
//-----------------------------------------
function prepare  num($data)
        {$r=ereg  replace('[^0-9]','',$data);
        if (($r=='')||(strlen($r)>12))$r=0;
        return $r;}
//-----------------------------------------
function prepare  hex($data)
        {$r=ereg  replace('[^0-9a-fA-F]','',$data);
        return $r;}
```

Также все строковые параметры (кроме паролей) следует обрабатывать функцией *htmlspecialchars* [1], следует обратить внимание на второй параметр функции – по умолчанию функция не заменяет кавычки. Исходя из вышесказанного, переданный логин следует обработать следующим образом: ***$ POST['login']=prepare login(htmlspecialchars($ POST['login'],ENT QUOTES))***. Все передаваемые скрипту параметры лучше проверять вначале, а не непосредственно перед использованием, в таком случае уменьшается вероятность, что параметр останется по ошибке не проверенным[2]. Использование функции *htmlspecialchars* позволяет избежать вставки пользователем html-тегов и/или скриптов (например, JavaScript) в гостевые книги, форумы и другие страницы, формируемые из данных, переданных пользователями.

Не следует считать, что пользователь будет работать только с интерфейсом сайта. Например, в реализованном форуме у пользователей имеется возможность редактировать некоторые сообщения, рядом с такими сообщениями располагается ссылка для редактирования сообщения (в качестве параметра передается id сообщения), для сообщений, недоступных для редактирования, такой ссылки нет. Однако пользователь вполне может отредактировать в адресной строке URL и указать id нужного сообщения. Поэтому, в функции редактирования сообщений выполняются проверки: может ли пользователь редактировать сообщения, доступно ли данное сообщение для редактирования.

Если необходимо предоставить пользователю возможность применять в сообщениях теги, лучше использовать bbcode[2]. При этом руководствоваться принципом «что не разрешено, то запрещено», при этом не следует просто заменять символы [,] на <,>, а выполнять замену целого тега, например [b] – на <b>

Разработанный сайт соответствует всем вышеизложенным рекомендациям, кроме того реализованы следующие функции: назначение прав пользователям; возможность редактирования, удаления сообщений; модерирование форума; поддержка bb-тегов; проверка правильности использования тегов; наличие скрытых тем и сообщений; главная страница выполнена как тема форума и может редактироваться всеми аутентифицированными пользователями.

Планируется: реализовать протоколирование редактирования сообщений с возможностью отката; расширить список поддерживаемых bb-тегов; реализовать загрузку файлов на сервер пользователями; реализовать отображение информации о пользователях, в том числе «присутствие на сайте»; доработать и внедрить поддержку пользовательских стилей.

Литература

[1] Кузнецов М.В. PHP 5. Практика разработки Web-сайта / М.В. Кузнецов, И.В. Симдянов, С.В. Голышев. – СПБ.: БХВ-Петербург, 2005. – 960 с.

[2] Фленов М.Е. PHP глазами хакера. – СПб.: БХВ-Петербург, 2005 – 304 с.